ЛЕКЦИЯ 10

Изучение выявления Фишинга моделями машинного обучения

КЛАССИФИКАЦИЯ

Алгоритмы машинного обучения:

- Дерево решений (Decision tree);
- Случайный лес (Random forest);
- XGBoost;
- CatBoost;
- AdaBoost

ЭТАПЫ ВЫЯВЛЕНИЯ ФИШИНГА С ПОМОЩЬЮ МАШИННОГО ОБУЧЕНИЯ

Сбор данных
Источники данных:
•**PhishTank, OpenPhish, APWG** – базы фишинговых сайтов
•**VirusTotal, URLhaus** – списки вредоносных URL
•**Лог-файлы прокси-серверов и веб-браузеров**
•**DNS-запросы и сетевой трафик**
Признаки фишинговых сайтов:
•**Длина URL** (фишинговые сайты часто используют длинные ссылки)
•**Использование IP-адресов в URL**
•**Подозрительные домены** (.tk, .ml, .ga, .cf, .gq)
•**Подменные символы в домене** (например, g00gle.com, paypa1.com)
•**HTTPS vs. HTTP** (многие фишинговые сайты не используют HTTPS)
•**Перенаправления (редиректы)**
•**Избыточное использование вложенных <iframe>**
•**Частота встречаемости домена в WHOIS**

ЭТАПЫ ВЫЯВЛЕНИЯ ФИШИНГА С ПОМОЩЬЮ МАШИННОГО ОБУЧЕНИЯ

```
phishing
```

	URL	Label
0	nobell.it/70ffb52d079109dca5664cce6f317373782/...	bad
1	www.dghjdgf.com/paypal.co.uk/cycgi-bin/webscrc...	bad
2	serviciosbys.com/paypal.cgi.bin.get-into.herf....	bad
3	mail.printakid.com/www.online.americanexpress....	bad
4	thewhiskeydregs.com/wp-content/themes/widescre...	bad
...	...	...
539341	paulasalamanca.com/g7cberv	bad
539342	trustcarts.com/g7cberv	bad
539343	rndled.com/g7cberv	bad
539344	rktest.net/g7cberv	bad
539345	unoldontal.com/g7cberv	bad

539346 rows × 2 columns

ЭТАПЫ ВЫЯВЛЕНИЯ ФИШИНГА С ПОМОЩЬЮ МАШИННОГО ОБУЧЕНИЯ

Предобработка данных

Перед применением алгоритмов машинного обучения данные должны быть очищены и преобразованы.

Действия:

- **Очистка данных** (удаление дубликатов, исправление ошибок)
- **Токенизация URL** (разделение домена, поддоменов, пути)
- **One-Hot Encoding** (преобразование категориальных признаков)
- **Нормализация числовых признаков** (например, длина URL)
- **Преобразование данных с помощью TF-IDF**

ЭТАПЫ ВЫЯВЛЕНИЯ ФИШИНГА С ПОМОЩЬЮ МАШИННОГО ОБУЧЕНИЯ

```
vectorizer = TfidfVectorizer(max_features = 100)
```

```
X = vectorizer.fit_transform(X.values)
```

```
X
```

```
<539346x100 sparse matrix of type '<class 'numpy.float64'>'
        with 1344974 stored elements in Compressed Sparse Row format>
```

```
X.shape
```

```
(539346, 100)
```

```
ros = RandomOverSampler(random_state=42)
```

```
X_resampled, y_resampled = ros.fit_resample(X, y)
```

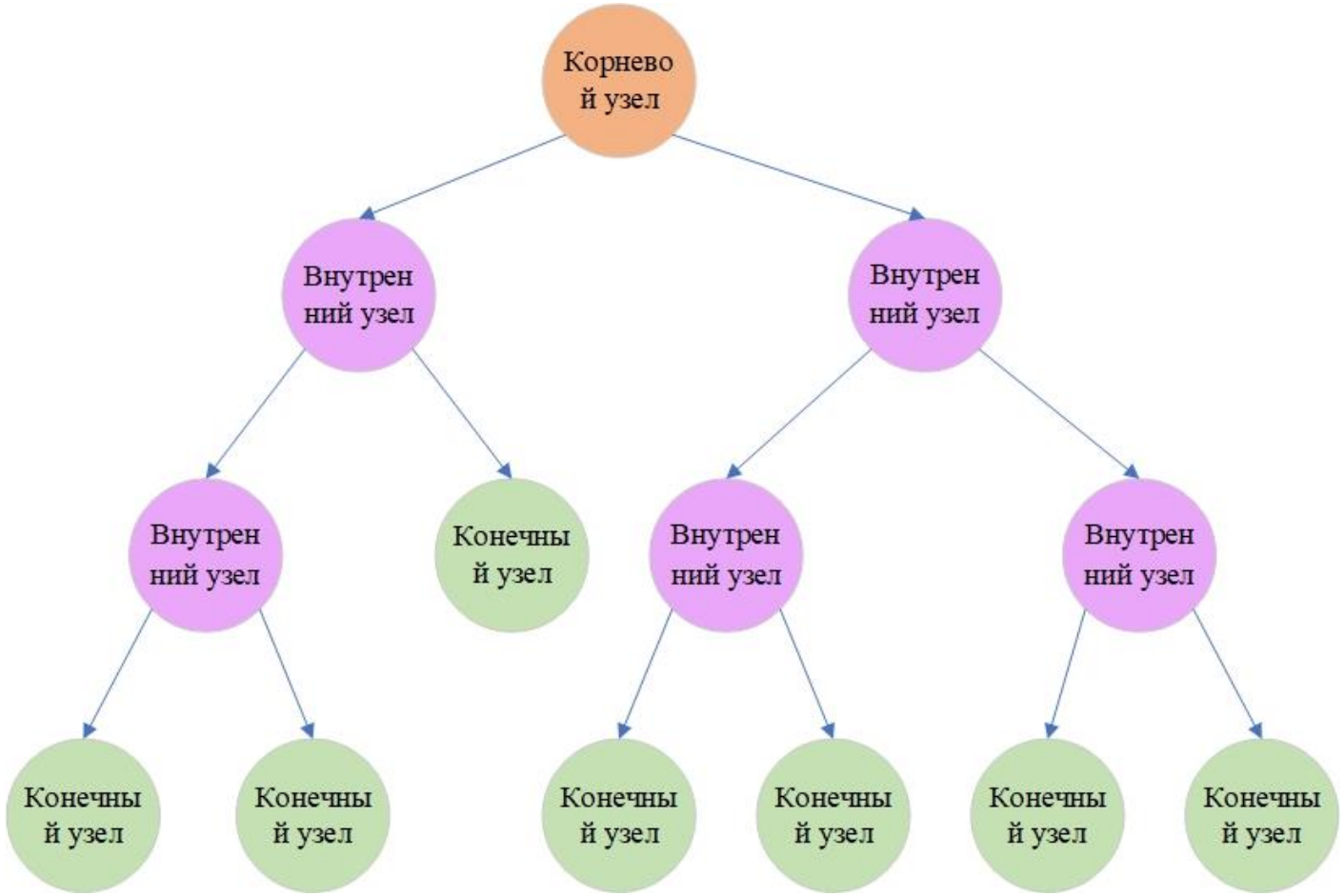
ЭТАПЫ ВЫЯВЛЕНИЯ ФИШИНГА С ПОМОЩЬЮ МАШИННОГО ОБУЧЕНИЯ

- **Разделение данных на обучающую и тестовую выборки**
- Данные разделяются следующим образом:
- **Обучающая выборка (Train Set)**: 70-80% данных
- **Тестовая выборка (Test Set)**: 20-30% данных
- Дополнительно можно использовать **кросс-валидацию (k-fold cross-validation)** для более точной оценки моделей.

ДЕРЕВО РЕШЕНИЙ

Дерево решений– метод обучения с учителем, который использует набор правил для принятия решений подобно тому, как человек принимает решения. В данном методе данные разделяются на подмножества в зависимости от определенных признаков, отвечая на определенные вопросы до тех пор, пока все точки данных не будут принадлежать определенному классу. Таким образом, образуется древовидная структура с добавлением узла для каждого вопроса. Первый узел является корневым узлом (root node). При классификации документов на первом этапе выбирается слово, и все документы, содержащие его, помещаются в одну сторону, а документы, не содержащие его, помещаются в другую сторону. В результате образуются два датасета. После этого в этих датасетах выбирается новое слово, и все предыдущие шаги повторяются. Так продолжается до тех пор, пока весь датасет не будет разделен и присвоен конечным узлам. Если в конечном узле все точки данных однозначно соответствуют одному и тому же классу, то класс узла точно определен. В случае смешанных узлов алгоритм присваивает данному узлу класс с наибольшим числом точек данных, относящихся к нему.

ДЕРЕВО РЕШЕНИЙ

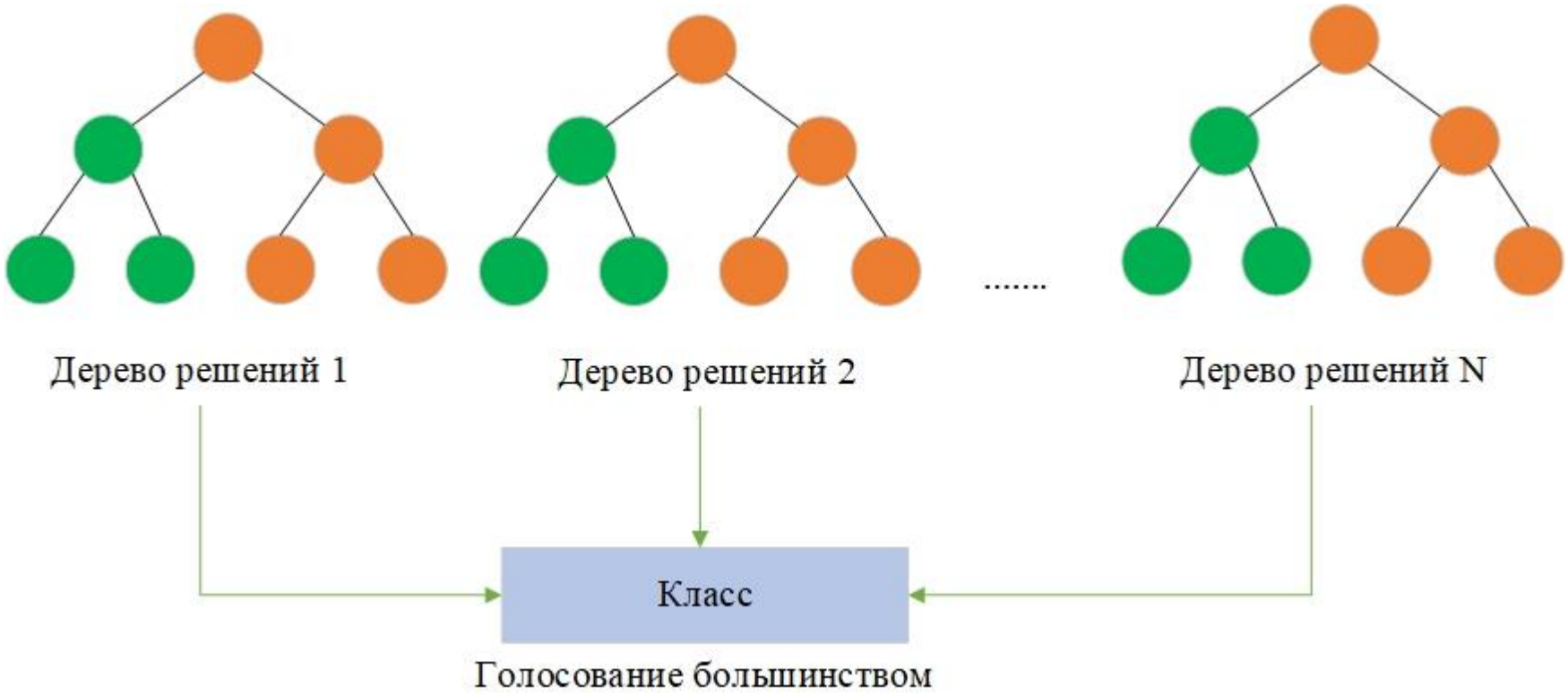


СЛУЧАЙНЫЙ ЛЕС

- Случайный лес– популярный алгоритм машинного обучения, основанный на концепции ансамблевого обучения. В данной концепции несколько классификаторов объединяются для улучшения производительности модели. Случайный лес состоит не из одного, а из множества деревьев решений. В задачах классификации каждый документ независимо кдассифицируется всеми деревьями. Класс документа определяется на основе наибольшего числа голосов среди всех деревьев.
- Алгоритм случайного леса имеет следующий ряд особенностей и преимуществ:

– Довольно быстро обучается.
– Эффективно обрабатывает датасеты с большим числом признаков.
– Выполняет предсказание данных с очень высокой точностью.
– Показывает хорошую эффективность даже при наличии большого числа пропусков данных.
– Хорошо обрабатываются как непрерывные, так и дискретные признаки.
– Обладает высокой масштабируемостью.

СЛУЧАЙНЫЙ ЛЕС



XGBOOST

- XGboost (eXtreme Gradient Boosting) – оптимизированный продвинутый алгоритм машинного обучения, использующий принцип бустинга. Он имеет хорошую производительность и решает большинство проблем регрессии и классификации. Использование ансамблевой техники подразумевает, что ошибки предыдущих шагов устраняются в новой модели. Отклонения прогнозов обученного ансамбля вычисляются на обучающем наборе на каждой итерации. Таким образом, оптимизация выполняется путем добавления новых древовидных прогнозов в ансамбль, уменьшая среднее отклонение модели. Эта процедура продолжается до тех пор, пока не будет достигнут требуемый уровень ошибки или критерий ранней остановки (максимальное количество деревьев или достижение заданной точности).

XGBoost

ЭТАПЫ ВЫЯВЛЕНИЯ ФИШИНГА С ПОМОЩЬЮ МАШИННОГО ОБУЧЕНИЯ

```
metrics_list = []
```

```
matrix_labels = ['phishing','not_phishing']
```

```
classifiers = [MultinomialNB(), LogisticRegression(), DecisionTreeClassifier(criterion='gini', splitter='best', max_depth=None),
RandomForestClassifier(n_estimators = 10), xgb.XGBClassifier(random_state=42), CatBoostClassifier(task_type="GPU", devices='0'),
AdaBoostClassifier(n_estimators=10)]
```

```
k = 0
for i in classifiers:
    i.fit(x_train, y_train)
    y_pred = i.predict(x_test)
    accuracy = accuracy_score(y_test, y_pred)
    precision = precision_score(y_test, y_pred)
    recall = recall_score(y_test, y_pred)
    f1 = f1_score(y_test, y_pred)
    fpr, tpr, threshold = metrics.roc_curve(y_test, y_pred)
    roc_auc = metrics.auc(fpr, tpr)
    metrics_list.append({'Accuracy': accuracy,
                         'Precision': precision,
                         'Recall': recall,
                         'F1-score': f1,
                         'fpr': fpr,
                         'tpr': tpr})

    print("Evaluation metrics of " + algorithms[k]+" algorithm: ")
    print('Accuracy: ', accuracy)
    print('Precision: ', precision)
    print('Recall: ', recall)
    print('F1-score: ', f1)

    k = k + 1
```

СПАСИБО ЗА ВНИМАНИЕ!!!